Week 4 and 5

Dashboarding Overview

© IBM Corporation. All rights reserved.

What you will learn

Explore the benefits of using interactive data applications to improve business performance

Identify different web-based dashboarding tools available in Python

Dashboard

- Real-time visuals simplify business moving parts
- Display Key Performance Indicators (KPI) for analysis
- Help businesses by providing the big picture

Best dashboards answer important business questions

Scenario

Review flight performance of US domestic flights

Yearly report items:

- Top 10 airline carriers in 2019
- Number of flights in 2019
- Number of travelers from California

Type one: Without dashboard

1.

Reporting_Airline	Flights
WN	13972
DL	10241
AA	9853
OO	8515
UA	6430
YX	3416
MQ	3368
B6	3170
OH	2969
AS	2783

2.

Month	Flights
1	6125
2	5578
3	6553
4	6282
5	6441
6	6583
7	6798
8	6753
9	6140
10	6533
11	6280
12	6550

3.

DestState	DistanceGroup	Flights
AK	10	4
AL	8	1
AR	7	1
AZ	2	361
AZ	3	220
AZ	4	9
CA	1	283
CA	2	2256
CA	3	36
CO	3	19
CO	4	401
CO	5	2
CT	11	2
FL	9	51
FL	10	90
FL	11	26
GA	8	114
GA	9	65
HI	10	168
HI	11	116
ID	2	7
ID	3	60
IL	7	169
IL	8	154
IN	8	16
KY	8	13
KY	9	5
LA	7	33
LA	8	14
MA	11	128

Type two: With dashboard

Web-based dashboarding

Dashboard tools

Bowtie

ipywidgets

Source: https://pyviz.org/dashboarding/

Recap

In this video, you learned that:

- Dashboard simplifies the dynamic aspects of the business.
- Data can be presented by using different types of dashboards.
- There are different types of dashboarding tools.

Additional Resources for Dashboards

For more information about Dashboards, visit the following links:

Python dashboarding tools

John Snow's data journalism

Completed

Go to next item

Like Dislike Report an issue

Introduction to Plotly

© IBM Corporation. All rights reserved.

What you will learn

Explore Plotly and its two sub-modules

Discover how to use Plotly graph objects and Plotly express for creating charts

Plotly: An overview

- Interactive, open-source plotting library
- Supports over 40 unique chart types
- Includes various types of charts
- Visualizations can be:
  - Displayed in Jupyter notebook
  - Saved to HTML files
  - Used in developing Python-built web applications

Plotly Sub-modules

- Plotly Graph Objects: Low-level interface to figures, traces, and layout

```
plotly.graph_objects.Figure
```

- Plotly Express: High-level wrapper

Using plotly.graph_objects

```
# Import required packages
import plotly.graph_objects as go
import plotly.express as px
import numpy as np

# Set random seed for reproducibility
np.random.seed(10)
x = np.arange(12)
# Create random y values
y = np.random.randint(50, 500, size=12)
```

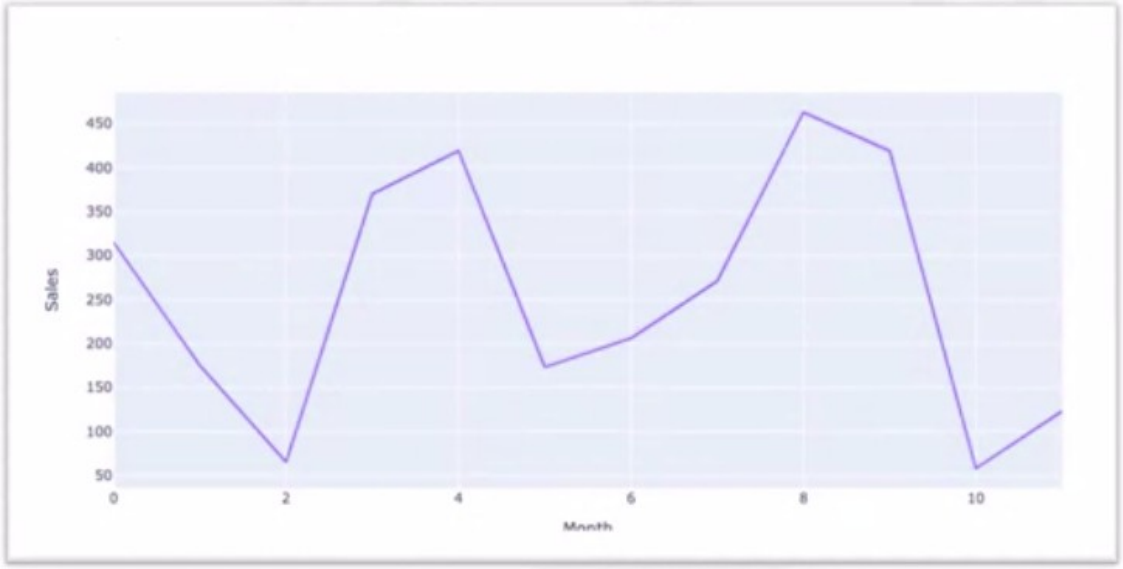
Using plotly.graph_objects

- Plotly.graph contains a JSON object
- Here, `go` is the plotly JSON object
- Chart is plotted by updating the values
- Figure is created by adding a trace

```
fig = go.Figure(data=go.Scatter(x=x, y=y))
```

Using plotly.graph_objects

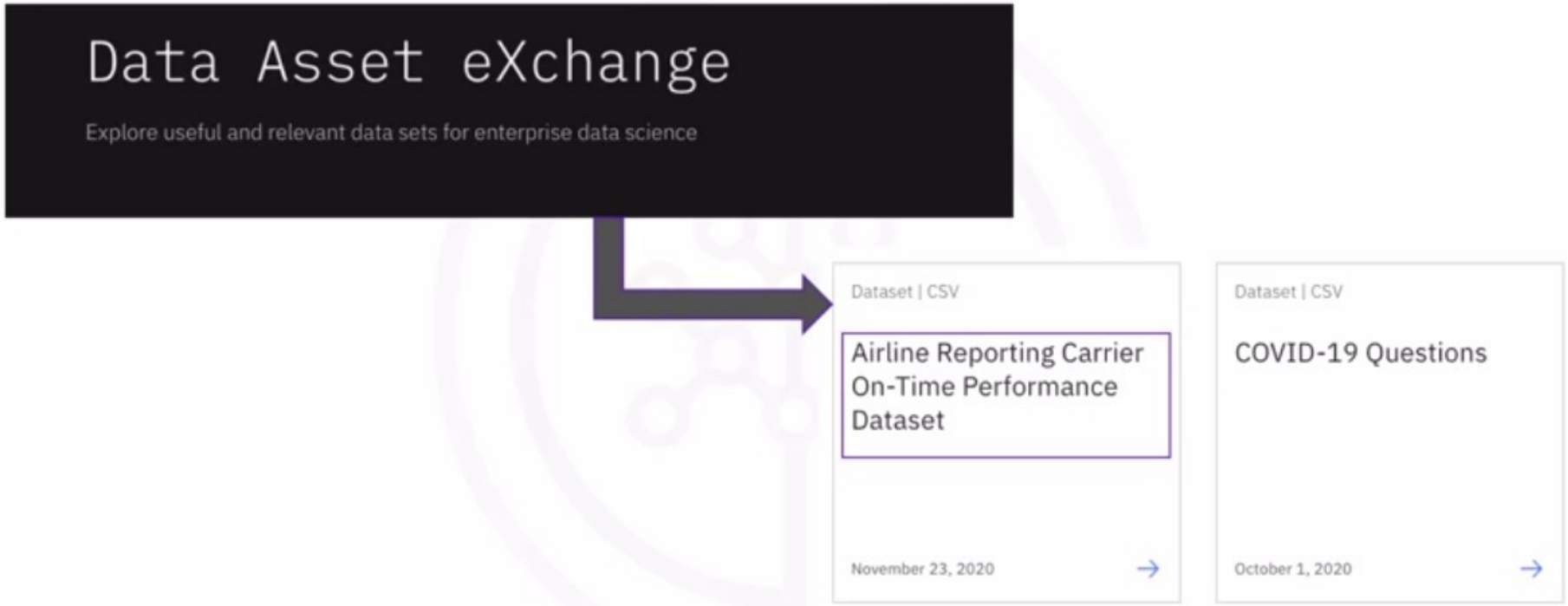
```
fig = go.Figure(data=go.Scatter(x=x, y=y))

fig.update_layout(title='Simple Line Plot', xaxis_title='Month', yaxis_title='Sales')
fig.show()
```



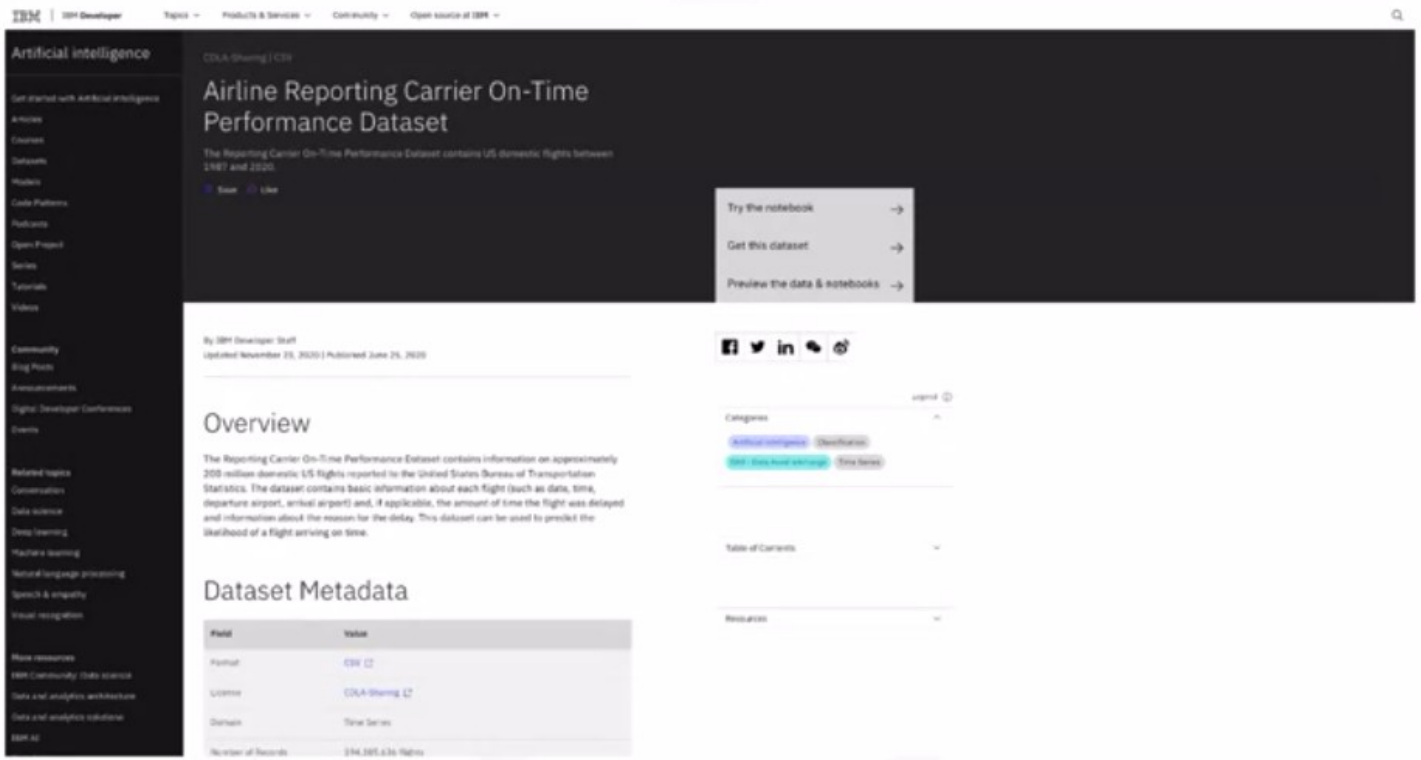
Using plotly.express

```
# Entire line chart can be created in a single command
fig = px.line(x=x, y=y, title='Simple Line Plot', labels=dict(x='Month', y='Sales'))
fig.show()
```

Dataset for Lab



Dataset for Lab



Recap

In this video, you learned that:

- Plotly is an interactive, open-source plotting library that supports over 40 unique chart types.
- Plotly graph objects is a low-level interface to figures, traces, and layout.
- Plotly express is a high-level wrapper for Plotly. It uses graph objects internally.

To learn more about using Plotly to create dashboards, explore

Plotly python

Plotly graph objects with example

Plotly express

API reference

Here are additional useful resources:

Plotly cheatsheet

Plotly community

Related blogs

Open-source datasets

Completed

Go to next item

Back

Practice Quiz: Creating Dashboards with Plotly

Practice Quiz • 10 min • 5 total points

Congratulations! You passed!

Grade received 100% **To pass** 80% or higher

Go to next item

1. Dashboards can provide real-time visuals. **1 / 1 point**

- (•) True
- ( ) False

Correct
Correct! Dashboards can provide real-time visuals. With real-time visuals on the dashboard, understanding business moving parts becomes easy.

2. What is Dash? **1 / 1 point**

- ( ) A framework tool used for building software.
- ( ) A framework application used for creating statistical graphs.
- ( ) Dash is a Python framework tool used for building Plotly applications.

Back

Practice Quiz: Creating Dashboards with Plotly

Practice Quiz • 10 min • 5 total points

2. What is Dash? **1 / 1 point**

- ○ A framework tool used for building software.
- ○ A framework application used for creating statistical graphs.
- ○ Dash is a Python framework tool used for building Plotly applications.
- ◉ Dash is a Python framework for building web analytic applications.

Correct

Correct! Dash is a Python framework for building web analytic applications.

3. What is Plotly? **1 / 1 point**

- ○ An interactive coding library.
- ◉ Plotly is an interactive, open-source plotting library that supports over 40 unique chart types.
- ○ An interactive chart that displays statistical data.
- ○ An interactive open-source code.

Correct

Correct! Plotly is an interactive, open-source plotting library that supports over 40 unique chart types.

Back

Practice Quiz: Creating Dashboards with Plotly

Practice Quiz • 10 min • 5 total points

4. A Callback function is a Python function that is automatically called by Dash whenever an input component's property changes. **1 / 1 point**

- (●) True
- ( ) False

Correct
Correct! A Callback function is a Python function that is automatically called by Dash whenever an input component's property changes. Callback functions are decorated with `@app.callback` decorator.

5. What two functions are required to create a callback? **1 / 1 point**

- ( ) Input and Library
- ( ) Output and Library
- ( ) Input and Outlet
- (●) Output and Input

Correct
Correct! The two functions required to create a callback are Output and Input.

Skills Network

Introduction to Dash

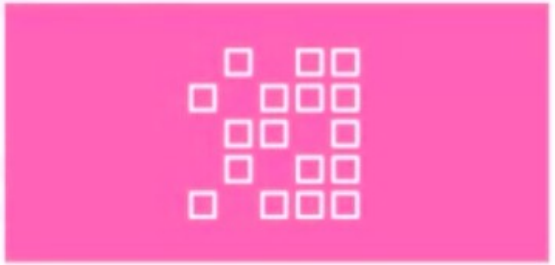
IBM

Introduction to Dash

© IBM Corporation. All rights reserved.

What you will learn

Explain what Dash is and its uses



Describe what Dash HTML components are

Dash – An Overview

- Open-source User Interface Python library from Plotly
- Easy to build GUI
- Declarative and Reactive
- Rendered in a web browser and can be deployed to servers
- Inherently cross-platform and mobile ready

Dash Components

- Core components

```
import dash_core_components as dcc
```

- HTML Components

```
import dash_html_components as html
```

Core Components

- Higher-level components that are interactive and are generated with JavaScript, HTML, and CSS through the React.js library

- Example: Creating a slider, input area, check items, datepicker, and other components

HTML Components

- Components for every HTML tag

- Keyword arguments describe the HTML attributes like style, className, and id

Recap

In this video, you learned that:

- Dash is an Open-Source User Interface Python library for creating reactive, web-based applications.
- It is easy to build Graphical User Interfaces using Dash as it abstracts all technologies required to make the applications.
- There are two components of Dash: Core and HTML components.
- The dash_core_components describe higher-level interactive components generated with JavaScript, HTML, and CSS through the React.js library.
- The dash_html_components library has a component for every HTML tag.

Additional Resources for Dash

To learn more about Dash, explore

Complete dash user guide

Dash core components

Dash HTML components

Dash community forum

Related blogs

Completed

Go to next item

Make Dashboards Interactive

© IBM Corporation. All rights reserved.

What you will learn

Describe the Callback function

Determine how to connect core and HTML components using Callbacks

Dash: Callbacks

- The Callback function is a python function
- It is automatically called by Dash

Decorator

```
@app.callback
```

Dash: Callback function

```
def callback_function:
    .....
    .....
    .....
    return some_result
```

Dash: Callback function

```
def callback_function:
    .....
    .....
    .....
    return some_result
```

@app.callback(Output, Input)

Dash: Callback function

Two parameters:

- **Output**

It sets results returned from the callback

- **Input**

It is provided to the callback function

Callback with one input

```
# Import required packages
import pandas as pd
import plotly.express as px
import dash
import dash_html_components as html
import dash_core_components as dcc
from dash.dependencies import Input, Output
# Read the data
airline_data = pd.read_csv('airline_2m.csv',
                           encoding = "ISO-8859-1",
                           dtype={'Div1Airport': str,
                           'Div1TailNum': str,
                           'Div2Airport': str,
                           'Div2TailNum': str})
```

Callback with one input

```
app = dash.Dash()
# Design dash app layout
app.layout = html.Div(children=[ html.H1('Airline Dashboard',
                                 style={'textAlign': 'center', 'color': colors['text'],
                                 'font-size': 40}),

                                 html.Div(["Input: ", dcc.Input(id='input-yr', value='2010',
                                 type='number', style={'height':'50px', 'font-size': 35}),],
                                 style={'font-size': 40}),

                                 html.Br(),
                                 html.Br(),
                                 html.Div(dcc.Graph(id='bar-plot')),
                                 ])
```

Callback with one input

```
@app.callback( Output(component_id='bar-plot', component_property='figure'),
               Input(component_id='input-yr', component_property='value'))
def get_graph(entered_year):
    # Select data
    df = airline_data[airline_data['Year']==int(entered_year)]
    # Top 10 airline carrier in terms of number of flights
    g1 = df.groupby(['Reporting_Airline'])['Flights'].sum().nlargest(10).reset_index()
    # Plot the graph
    fig1 = px.bar(g1, x='Reporting_Airline', y='Flights', title='Top 10 airline carrier in
                  year ' + str(entered_year) + ' in terms of number of flights')
    fig1.update_layout()
    return fig1
if __name__ == '__main__':
    app.run_server(port=8002, host='127.0.0.1', debug=True)
```

Callback with one input

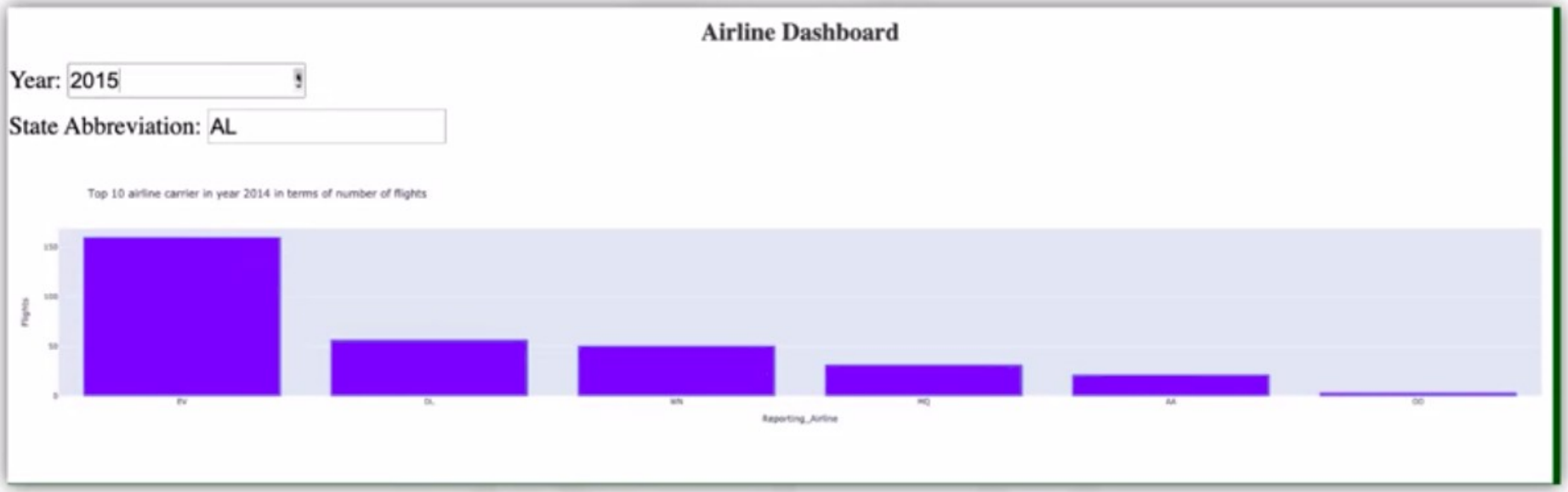
Callback with two inputs

```
app = dash.Dash()
# Design dash app layout
app.layout = html.Div(children=[ html.H1('Airline Dashboard', style={'textAlign': 'center',
                                 'color': colors['text'], 'font-size': 40}),
                                 html.Div(["Year: ", dcc.Input(id='input-yr', value='2010',
                                 type='number', style={'height':'50px', 'font-size': 35}),
                                 ], style={'font-size': 40}),
                                 html.Div(["State Abbreviation: ", dcc.Input(id='input-ab',
                                 value='AL', type='text', style={'height':'50px',
                                 'font-size': 35})], style={'font-size': 40}),
                                 html.Br(),
                                 html.Br(),
                                 html.Div(dcc.Graph(id='bar-plot')),
                                 ])
```

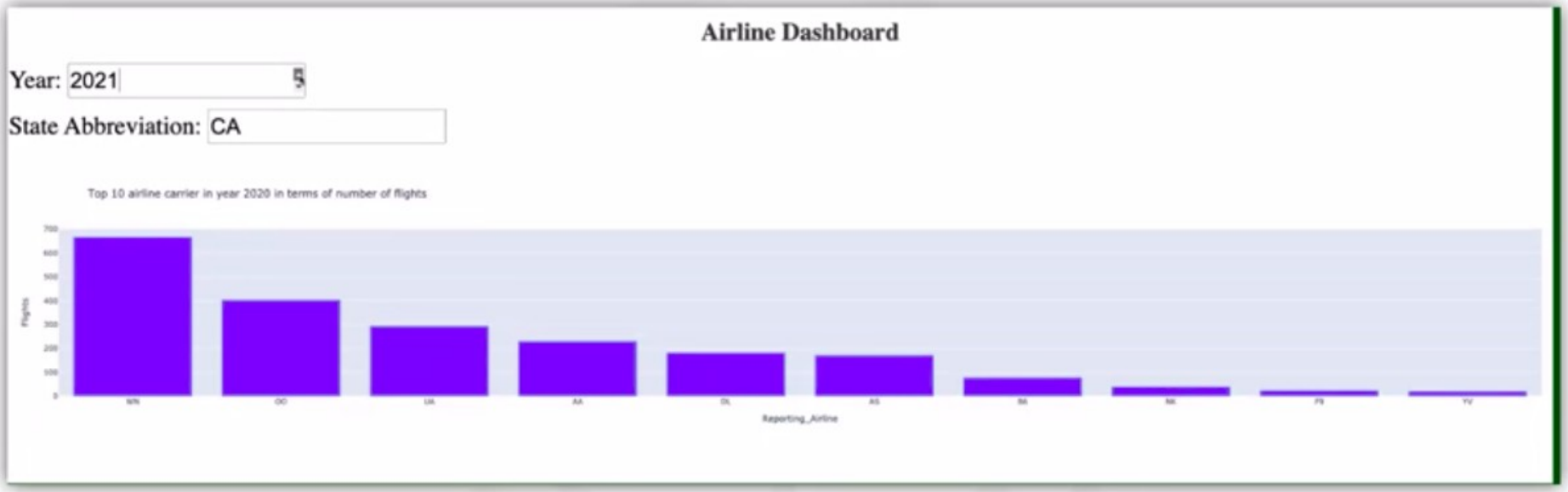
Callback with two inputs

```
@app.callback( Output(component_id='bar-plot', component_property='figure'),
               [ Input(component_id='input-yr', component_property='value'),
                 Input(component_id='input-ab', component_property='value')])

def get_graph(entered_year, entered_state):
    # Select data
    df = airline_data[ (airline_data['Year']==int(entered_year)) &
                       (airline_data['OriginState'] == entered_state)]
    # Top 10 airline carrier in terms of number of flights
    ……….
    fig1.update_layout()
    return fig1
if __name__ == '__main__':
app.run_server(port=8002, host='127.0.0.1', debug=True)
```

Callback with two inputs



Callback with two inputs



Recap

In this video, you learned that:

- A callback function is a python function that is automatically called by Dash.
- The @app.callback decorator decorates the callback function in order to tell Dash to call it whenever there is a change in the input component value.
- The callback function takes input and output components as parameters and performs operations to return the desired result for the output component.

Additional Resources for Interactive Dashboards

To learn more about making interactive dashboards in Dash, visit

Python decorators reference 1

Python decorators reference 2

Callbacks with example

Dash app gallery

Dash community components

Completed Go to next item

Walkthrough the Flight Delay Time Statistics Dashboard Lab

© IBM Corporation. All rights reserved.

What you will learn

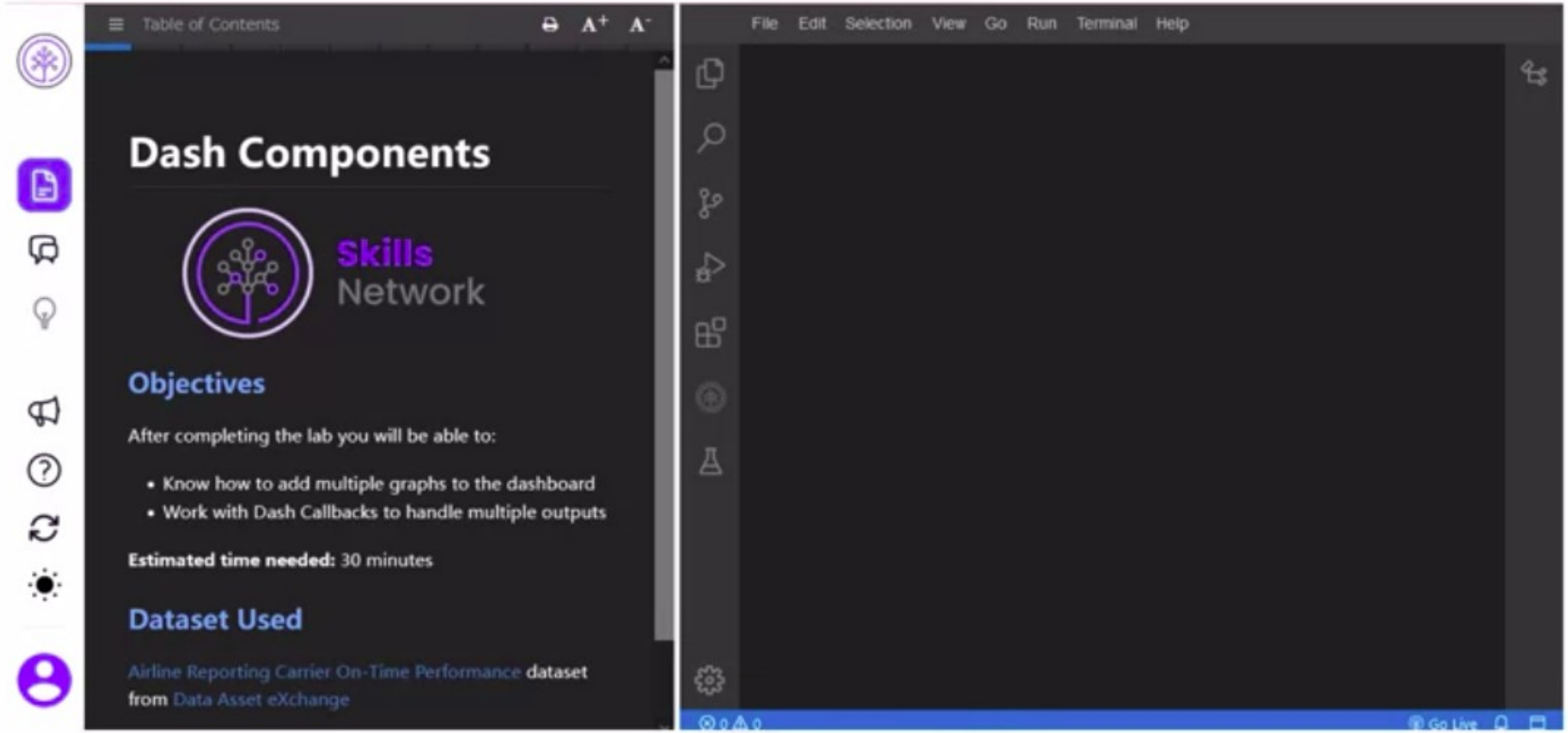
Describe the Skills Network Labs Cloud IDE environment

Analyze flight on-time data by creating line graphs with dashboard components

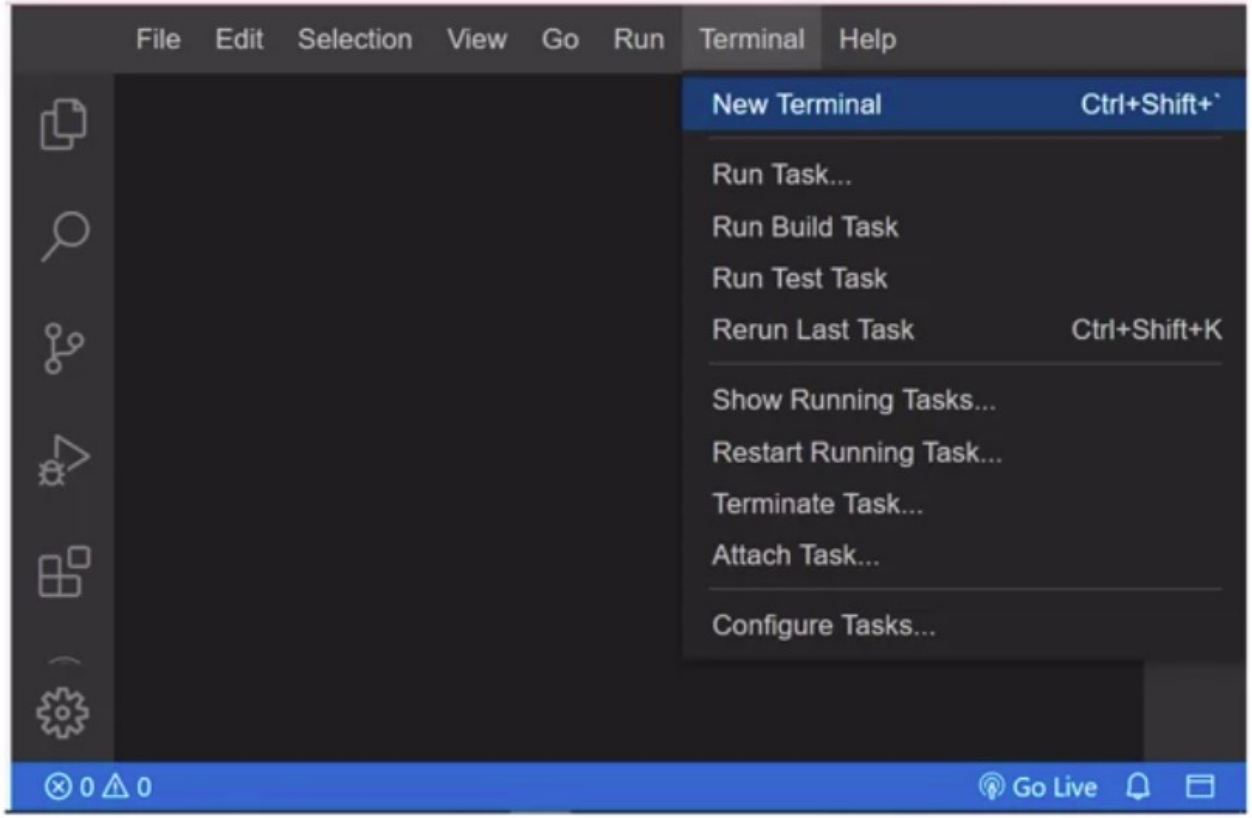
Create the layout of the dash application

List the steps to run and launch the application

Skills Network Labs (SN Labs) Cloud IDE environment

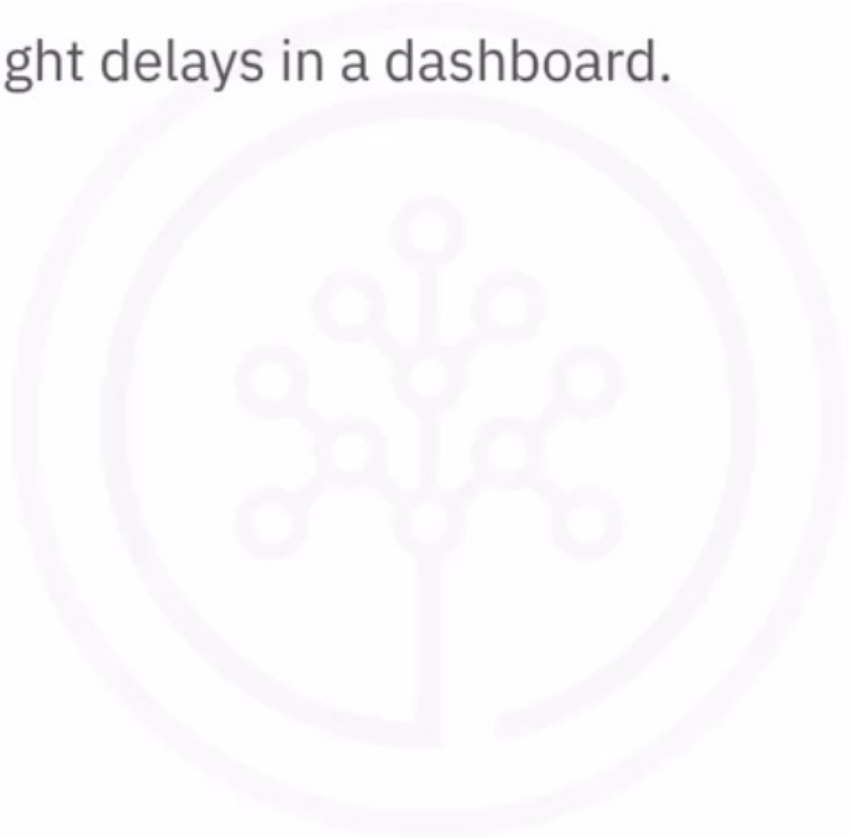


Open a new terminal



Install Python packages

You will analyze flight delays in a dashboard.



Dataset

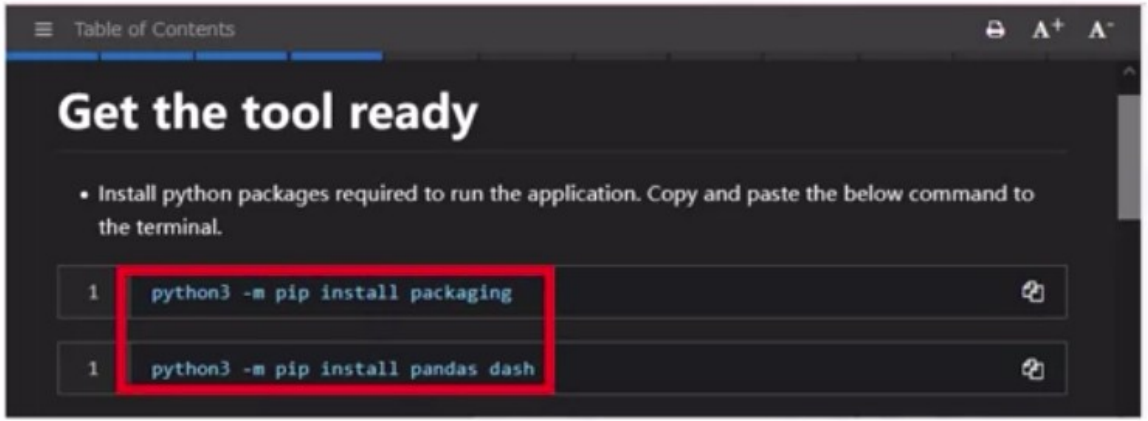
1. Airline reporting carrier on-time performance dataset
2. Information on approximately 200 million domestic US flights
3. Basic information about each flight: Time, date, departure and arrival airport, duration of flight delay, and so on

Install Python packages

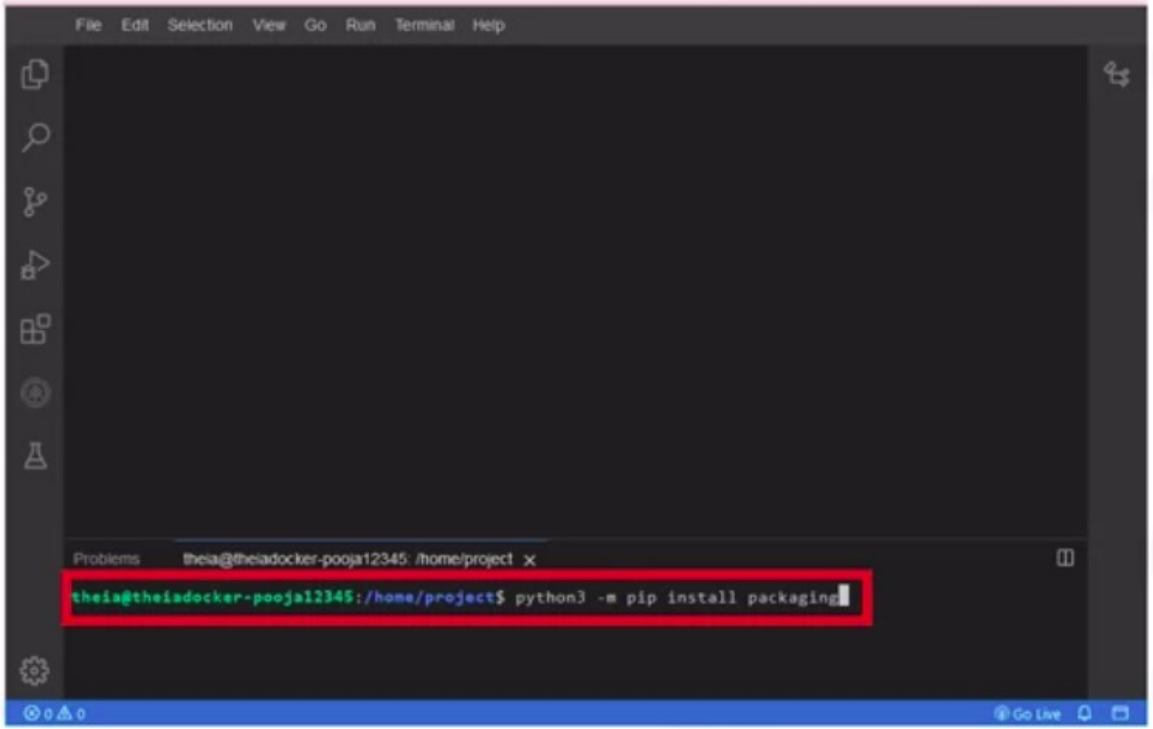
You will analyze flight delays in a dashboard.

- Monthly average carrier delay
- Monthly average weather delay
- Monthly average national air system delay
- Monthly average security delay
- Monthly average late aircraft delay by reporting airline for the given year.

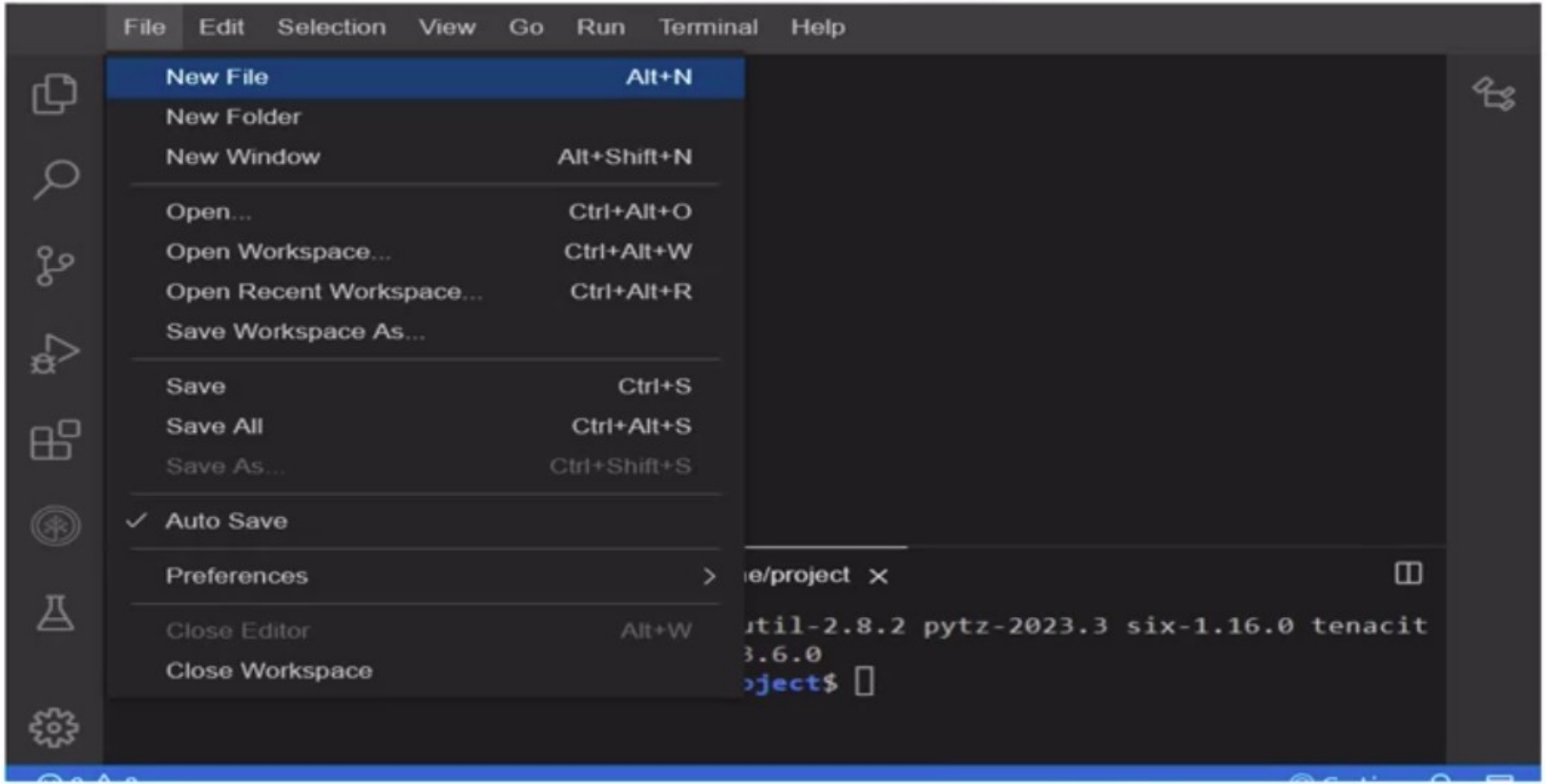
Install Python packages



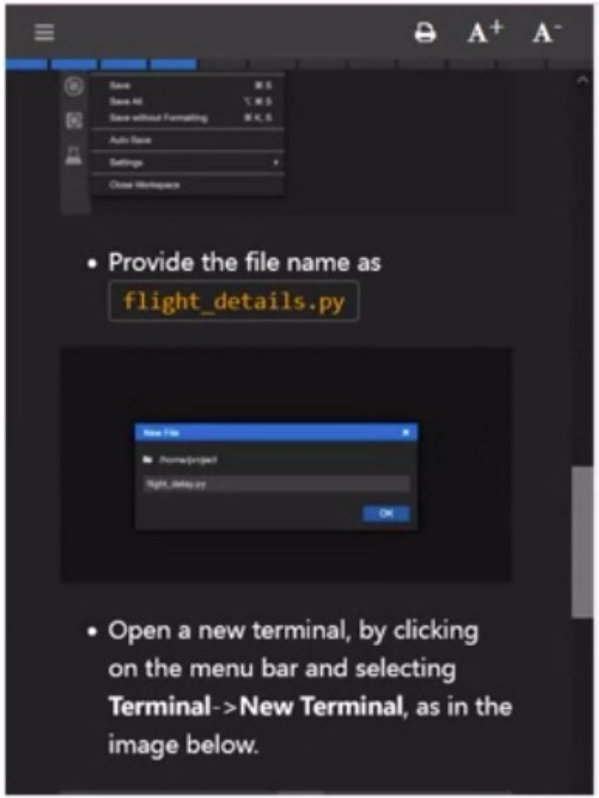
Install Python packages

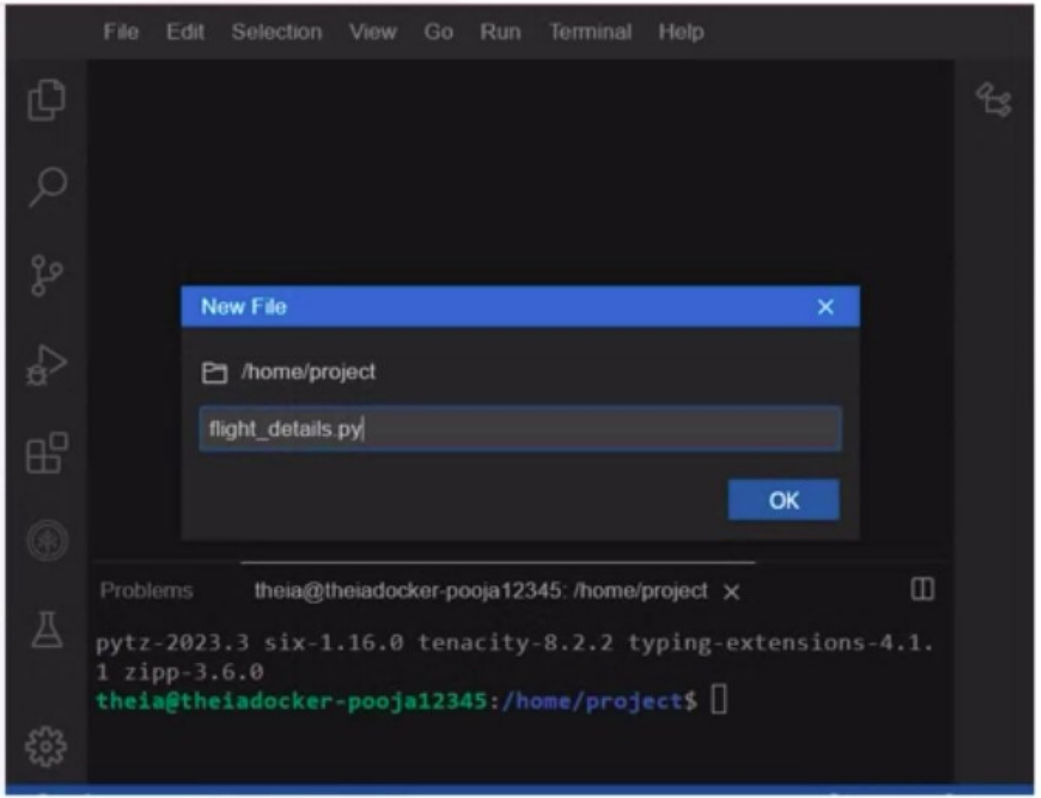


Create a new Python script

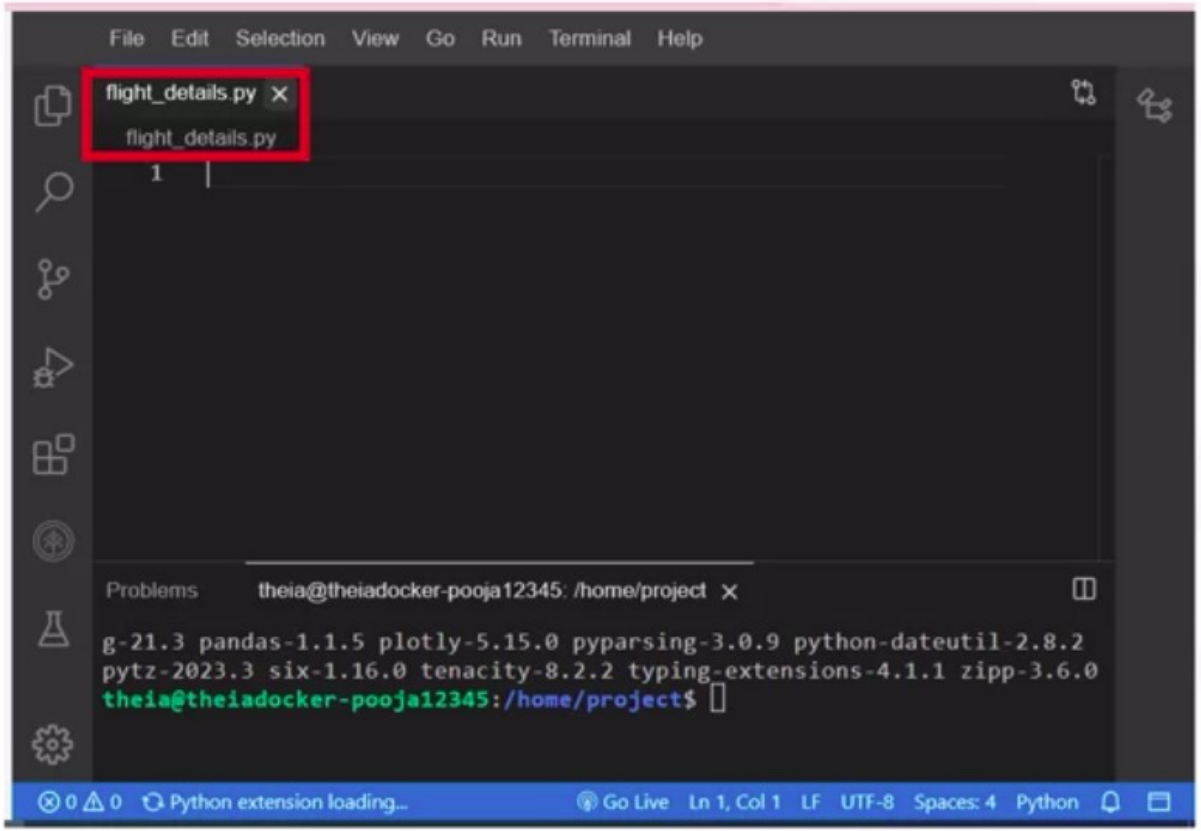


Create a new Python script

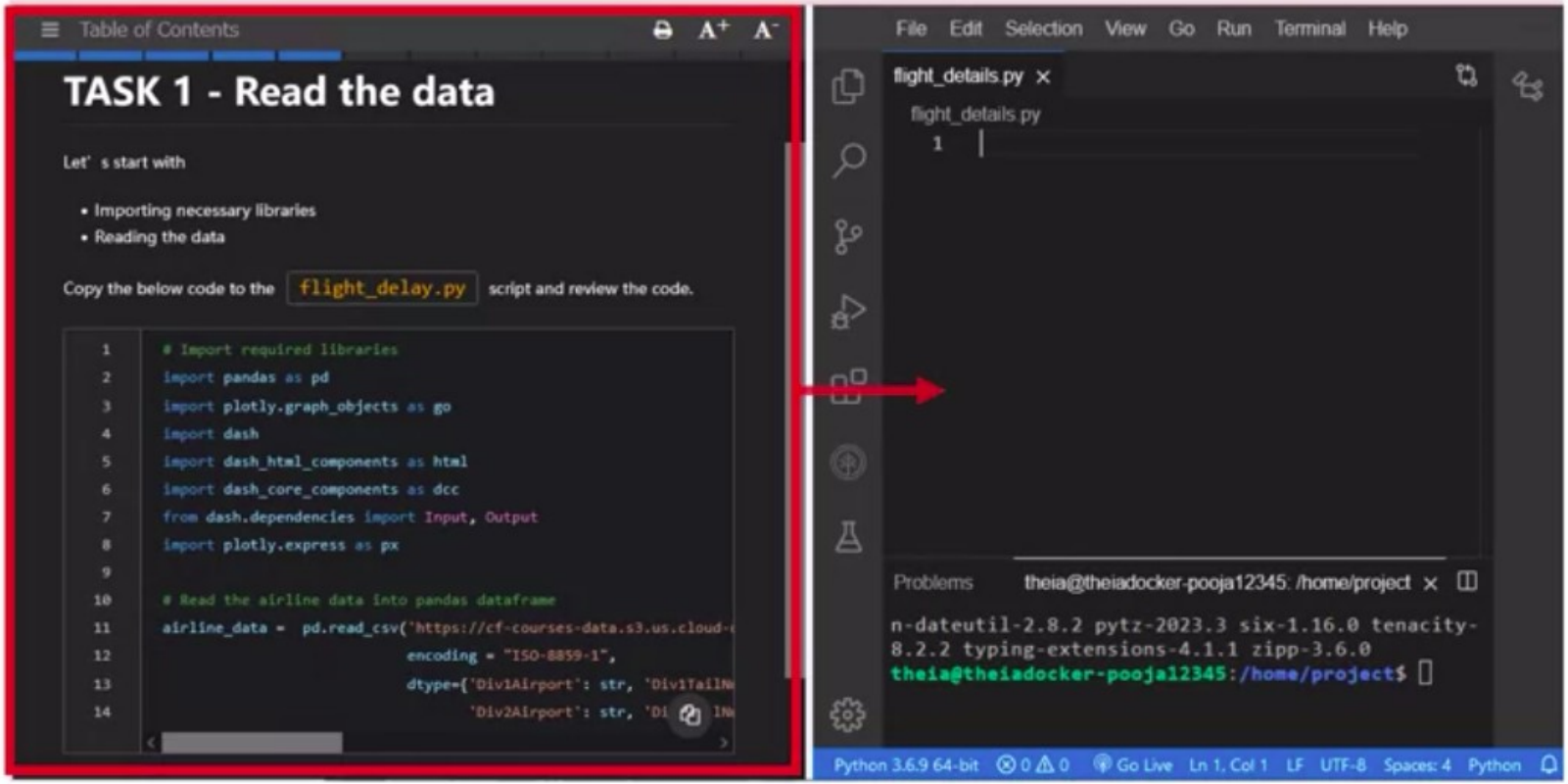




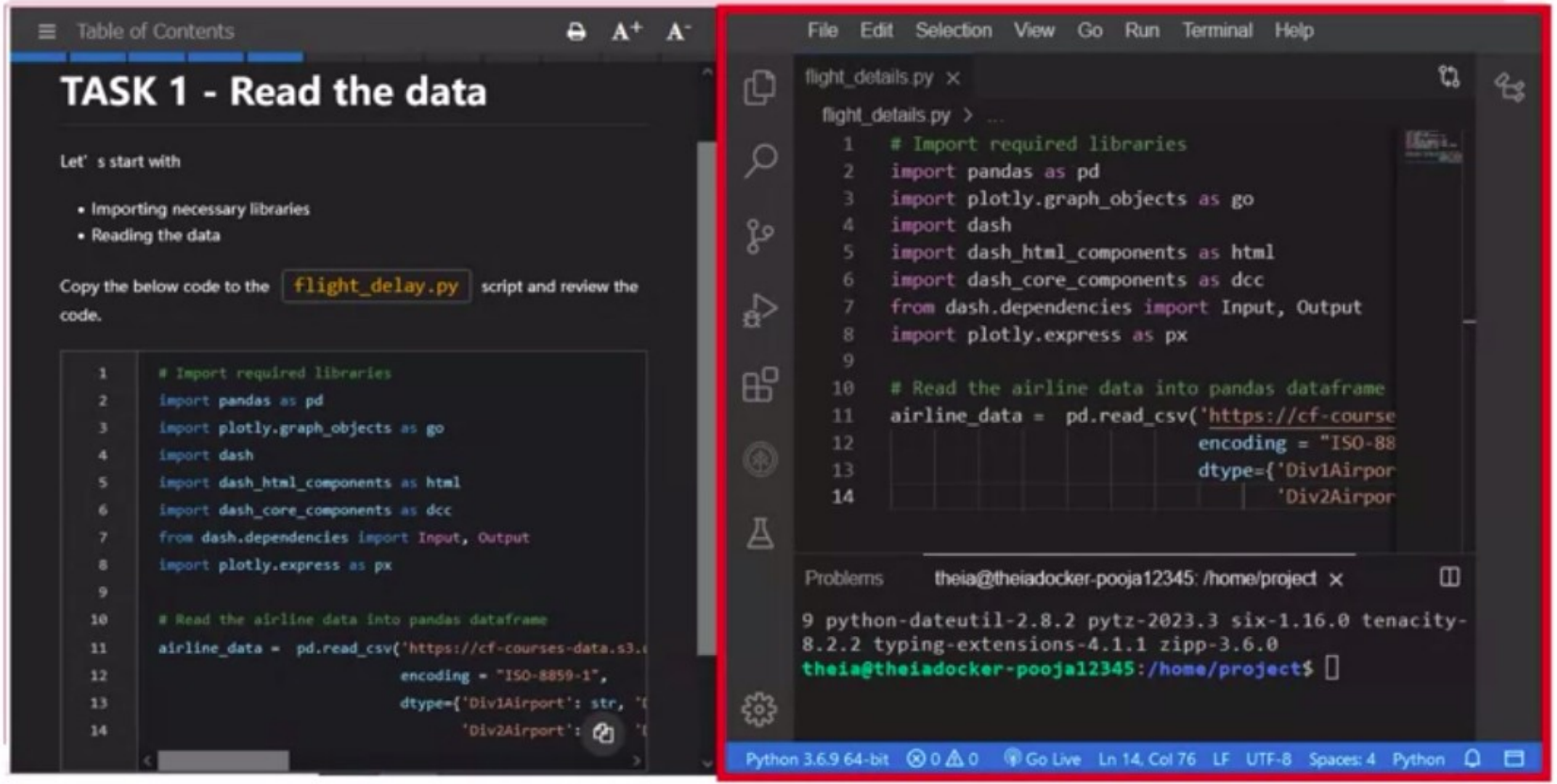
Create a new Python script



Import libraries and dataset



Import libraries and dataset



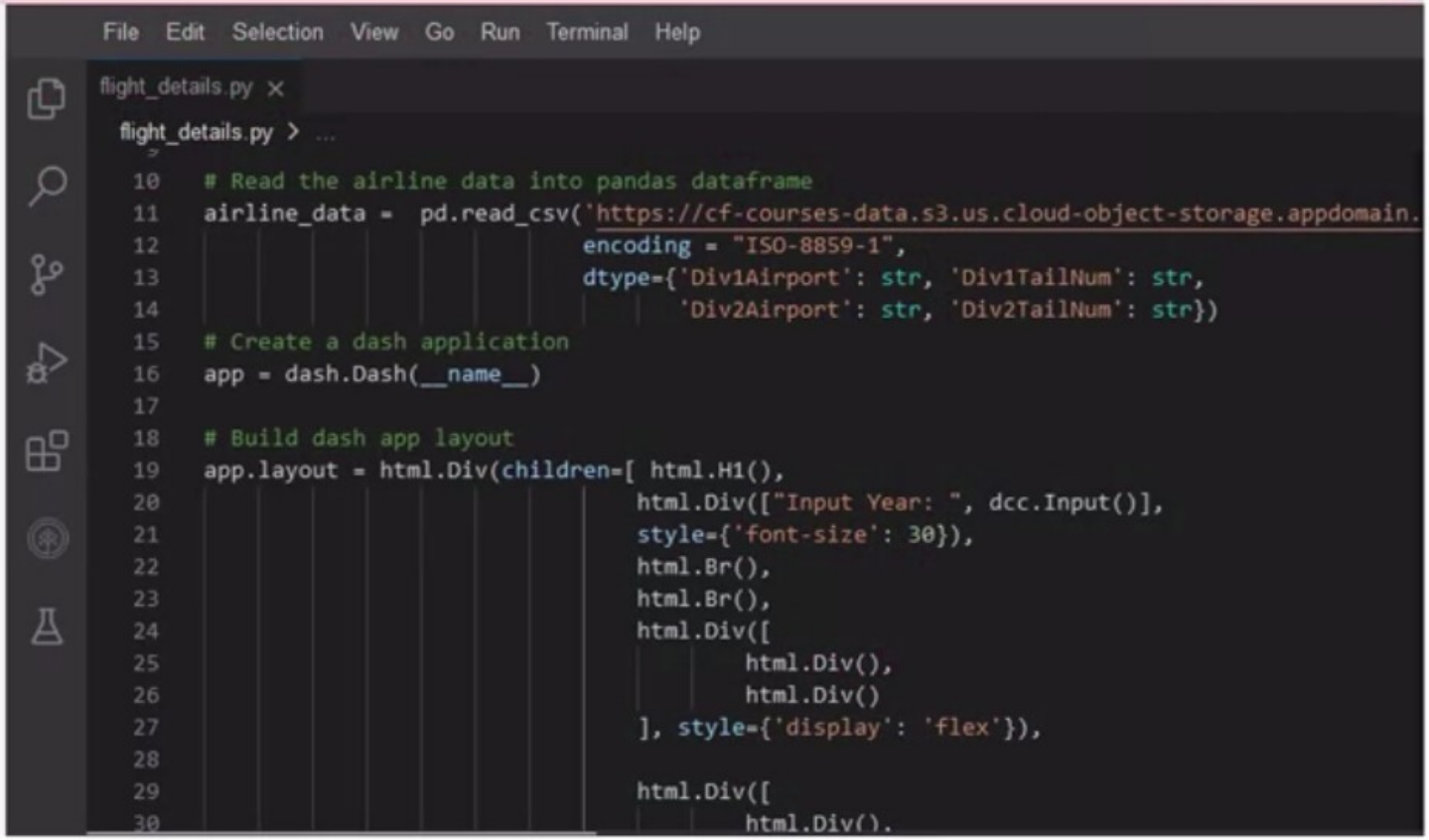
Create a skeleton for dash application

```
# Create a dash application
app = dash.Dash(__name__)
```

Create a skeleton for dash application

```
# Build dash app layout
app.layout = html.Div(children=[ html.H1(),
                                 html.Div(["Input Year: ", dcc.Input()],
                                 style={'font-size': 30}),
                                 html.Br(),
                                 html.Br(),
                                 html.Div([
                                         html.Div(),
                                         html.Div()
                                 ], style={'display': 'flex'}),

                                 html.Div([
                                         html.Div(),
                                         html.Div()
                                 ], style={'display': 'flex'}),

                                 html.Div(, style={'width':'65%'})
                                 ])
```

Create a skeleton for dash application



Include title and input

```
html.H1('Flight Delay Time Statistics',
                                    style={'textAlign': 'center', 'color':
'#503D36',
                                    'font-size': 30})
```

```
html.Div(["Input Year: ", dcc.Input(id='input-year', value='2010',
                        type='number', style={'height':'35px', 'font-
size': 30}),],
```

Include the graphs

```
html.Div([
    html.Div(dcc.Graph(id='carrier-plot')),
    html.Div(dcc.Graph(id='weather-plot'))
], style={'display': 'flex'})
```

Prepare data to plot

```
def compute_info(airline_data, entered_year):
    # Select data
    df =  airline_data[airline_data['Year']==int(entered_year)]
    # Compute delay averages
    avg_car =
df.groupby(['Month','Reporting_Airline'])['CarrierDelay'].mean().reset_index()
    avg_weather =
df.groupby(['Month','Reporting_Airline'])['WeatherDelay'].mean().reset_index()
    avg_NAS =
df.groupby(['Month','Reporting_Airline'])['NASDelay'].mean().reset_index()
    avg_sec =
df.groupby(['Month','Reporting_Airline'])['SecurityDelay'].mean().reset_index()
    avg_late =
df.groupby(['Month','Reporting_Airline'])['LateAircraftDelay'].mean().reset_ind
ex()
    return avg_car, avg_weather, avg_NAS, avg_sec, avg_late
```

Callback for real-time dashboard update

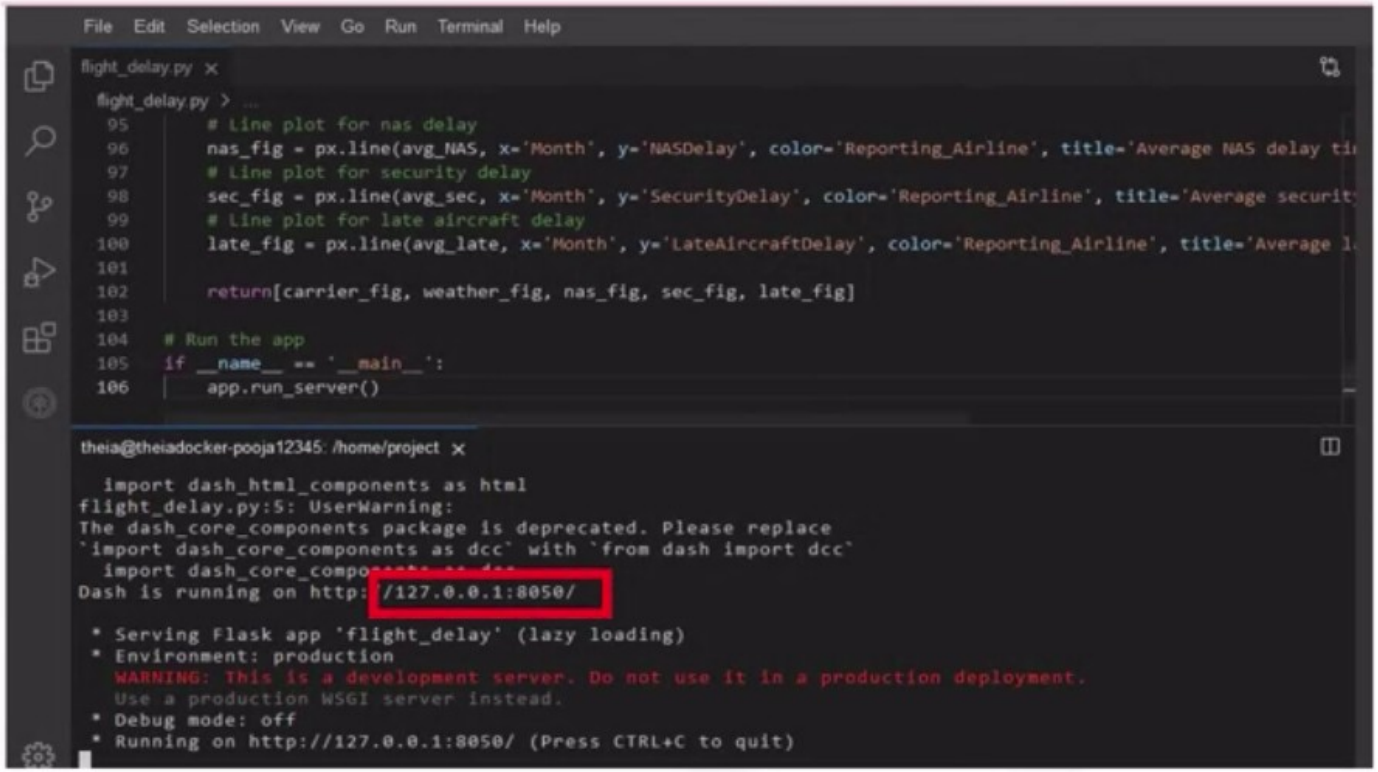
```
@app.callback( [
    Output(component_id='carrier-plot', component_property='figure'),
```

```
Input(component_id='input-year', component_property='value')
```

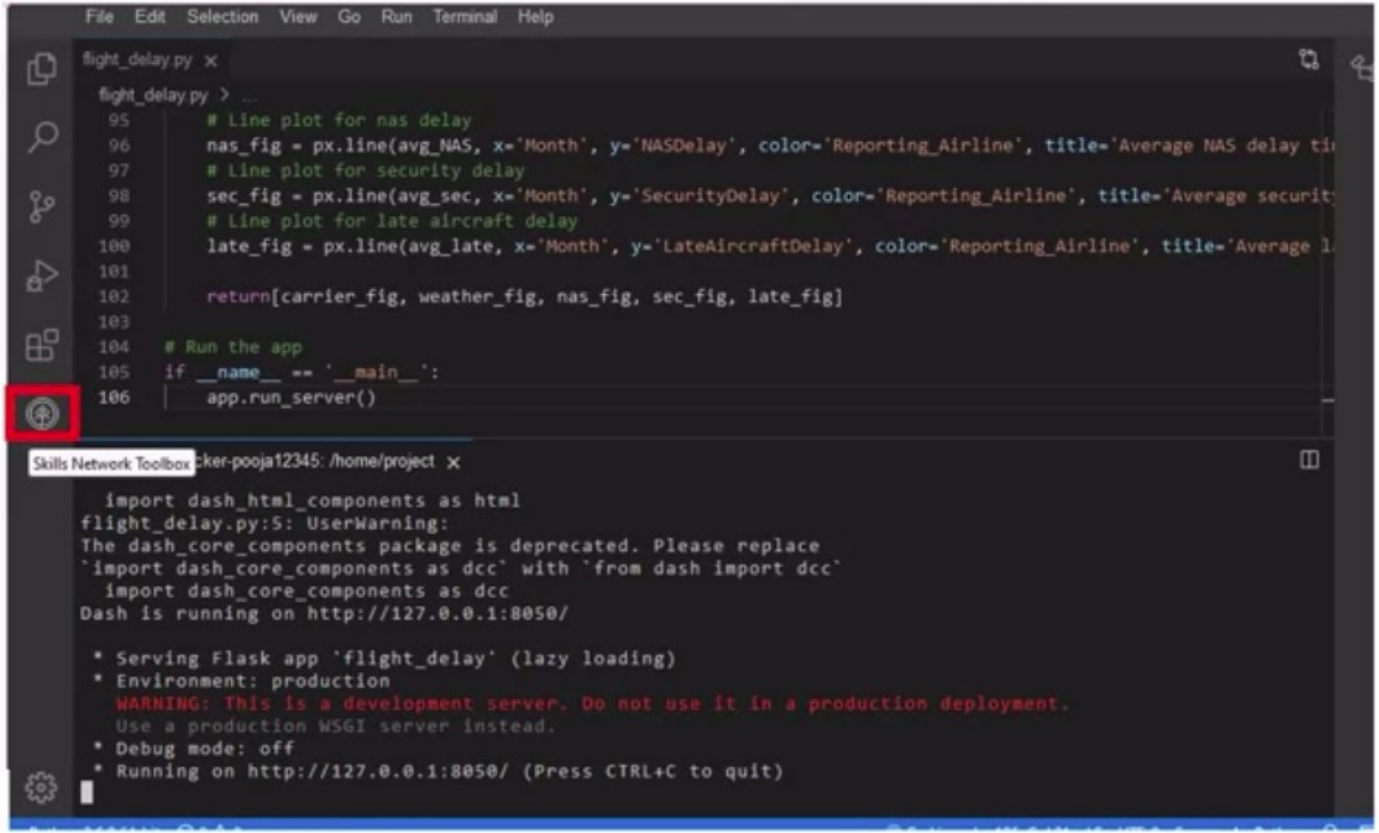
```
def get_graph(entered_year):

    # Compute required information for creating graph from the data
    avg_car, avg_weather, avg_NAS, avg_sec, avg_late =
compute_info(airline_data, entered_year)

    # Line plot for carrier delay
    carrier_fig = px.line(avg_car, x='Month', y='CarrierDelay',
color='Reporting_Airline', title='Average
```

```
if __name__ == '__main__':
    app.run_server()
```

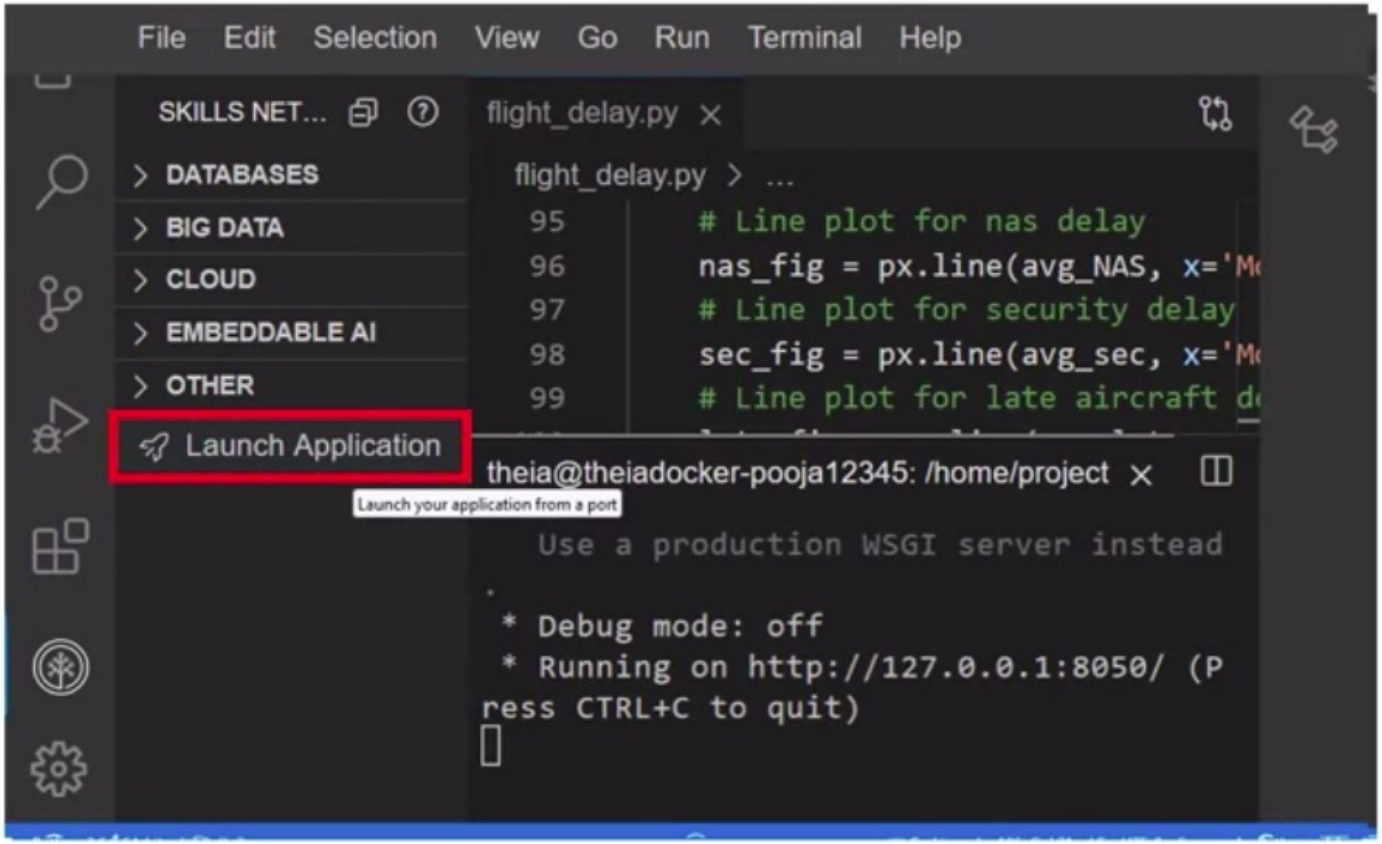
Launch application



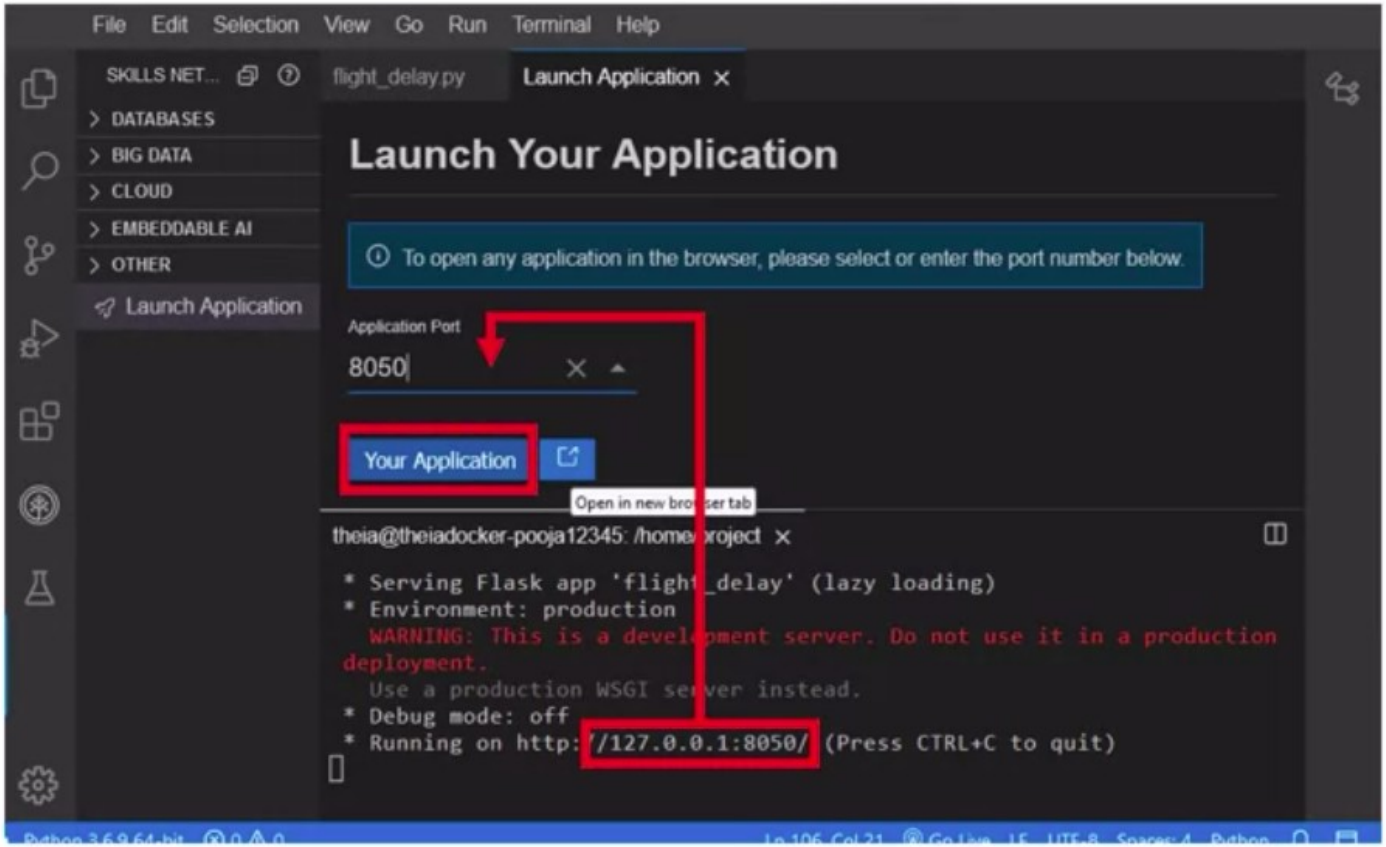
Launch application



Launch application



Launch application



Launch application

IBM

Recap

In this video, you learned

- We have walked through the lab where you developed a dashboard with Dash framework on Flight Delay Time Statistics

Back

Practice Quiz: Working with Dash

Practice Quiz • 10 min • 5 total points

Congratulations! You passed!

Grade received 100% **To pass** 80% or higher

Go to next item

1. True or False: Dashboard simplifies the dynamic aspects of the business. **1 / 1 point**

- (•) True
- ( ) False

Correct

Correct! With real-time visuals on the dashboard, understanding business moving parts becomes easy.

2. Fill in the blank. Plotly express is a _____ wrapper. **1 / 1 point**

- (•) High-level
- ( ) Low-level

False

Correct
Correct! With real-time visuals on the dashboard, understanding business moving parts becomes easy.

2. Fill in the blank. Plotly express is a _____ wrapper. **1 / 1 point**

- (●) High-level
- ( ) Low-level

Correct
Correct! Plotly Express is a high-level wrapper for Plotly. It is a recommended starting point for creating the most common figures provided by Plotly because of its simple syntax. It uses graph objects internally.

3. A Callback function is a Python function that is automatically called by _____ whenever an input component's property changes. **1 / 1 point**

- ( ) HTML
- ( ) Matplotlip
- (●) Dash

its simple syntax. It uses graph objects internally.

3. A Callback function is a Python function that is automatically called by _____ whenever an input component's property changes. **1 / 1 point**

- ○ HTML
- ○ Matplotlip
- ◉ Dash
- ○ Plotly

Correct
Correct! A Callback function is a Python function automatically called by Dash whenever an input component's property changes. Callback functions use the `@app.callback` decorator.

4. Which of the following is the correct way to add the callback decorator? **1 / 1 point**

- ○ @app.callback[Output(component_id='bar-plot', component_property='figure'), Input(component_id='input-yr', component_property='value')]
- ○ @app.callback(Output{component_id='bar-plot', component_property='figure'}, Input{component_id='input-yr', component_property='value'})

Correct
Correct! A Callback function is a Python function automatically called by Dash whenever an input component's property changes. Callback functions use the `@app.callback` decorator.

4. Which of the following is the correct way to add the callback decorator? **1 / 1 point**

- @app.callback[Output(component_id='bar-plot', component_property='figure'), Input(component_id='input-yr', component_property='value')]
- @app.callback(Output{component_id='bar-plot', component_property='figure'}, Input{component_id='input-yr', component_property='value'})
- (●) @app.callback(Output(component_id='bar-plot', component_property='figure'), Input(component_id='input-yr', component_property='value'))
- @app.callback{ Output{component_id='bar-plot', component_property='figure'}, Input{component_id='input-yr', component_property='value'}}

Correct
Correct! The correct way of adding a callback decorator is the following: @app.callback(Output(component_id='bar-plot', component_property='figure'), Input(component_id='input-yr,' component_property='value')).

5. The callback function takes input and output components as _____. **1 / 1 point**

- (●) Parameters

- (●) @app.callback(Output(component_id='bar-plot', component_property='figure'), Input(component_id='input-yr', component_property='value'))
- ( ) @app.callback{ Output{component_id='bar-plot', component_property='figure'}, Input{component_id='input-yr', component_property='value'}}

Correct
Correct! The correct way of adding a callback decorator is the following: @app.callback(Output(component_id='bar-plot', component_property='figure'), Input(component_id='input-yr,' component_property='value')).

5. The callback function takes input and output components as _____. **1 / 1 point**

- (●) Parameters
- ( ) Matplotlib
- ( ) Data
- ( ) Perimeter

Correct
Correct! The callback function takes input and output components as parameters and performs operations to return the desired result for the output component.

Congratulations! You have completed this module. At this point in the course, you know:

- Dash is an Open-Source User Interface Python library for creating reactive, web-based applications.
- It is easy to build Graphical User Interfaces using Dash as it abstracts all technologies required to make the applications.
- There are two components of Dash: Core and HTML components.
- The dash_core_components describe higher-level interactive components generated with JavaScript, HTML, and CSS through the React.js library.
- The dash_html_components library has a component for every HTML tag.
- A callback function is a python function that is automatically called by Dash whenever an input component's property changes.
- The @app.callback decorator decorates the callback function in order to tell Dash to call it whenever there is a change in the input component value.
- The callback function takes input and output components as parameters and performs operations to return the desired result for the output component.

Coursera | Online Courses | Final Assignment: Part 2 - | DV0101EN-Exercise-Final- | Auto Sales Dashboard Titl | Dash

127.0.0.1:8050

Update

Python Tutor co... | C Program to Ch... | Other bookmarks

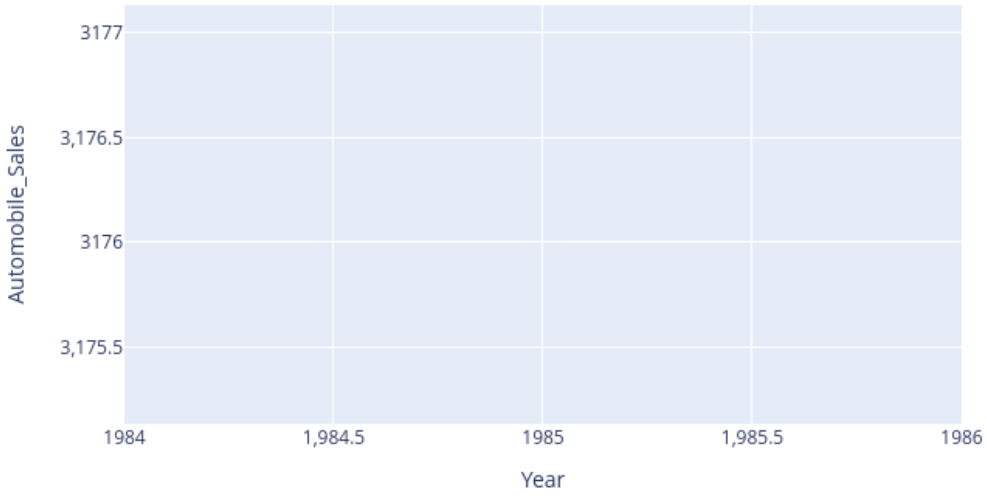
Automobile Sales Statistics Dashboard

Select Statistics:

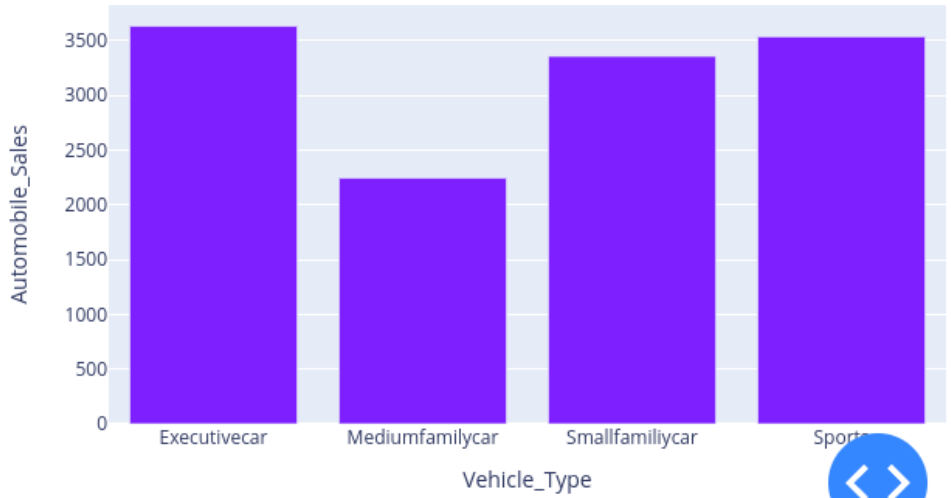
Yearly Statistics

1985

Yearly Automobile Sales



Average Vehicles Sold by Vehicle Type in the year 1985



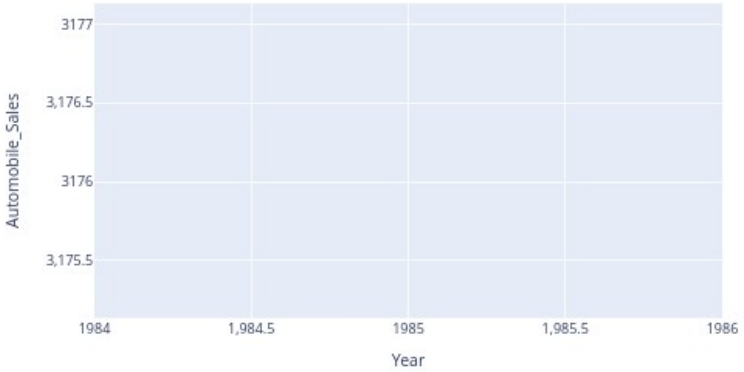
Automobile Sales Statistics Dashboard

Select Statistics:

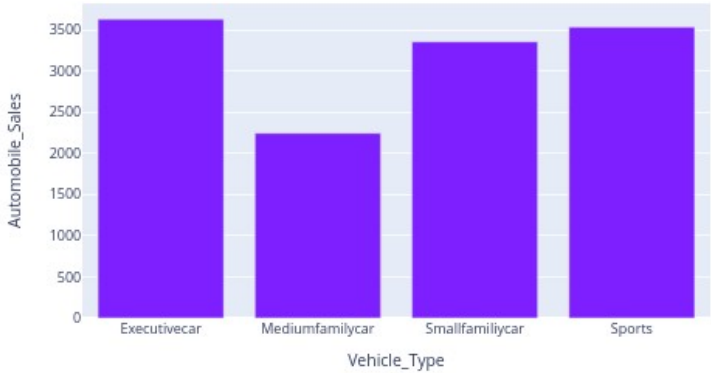
Yearly Statistics

1985

Yearly Automobile Sales

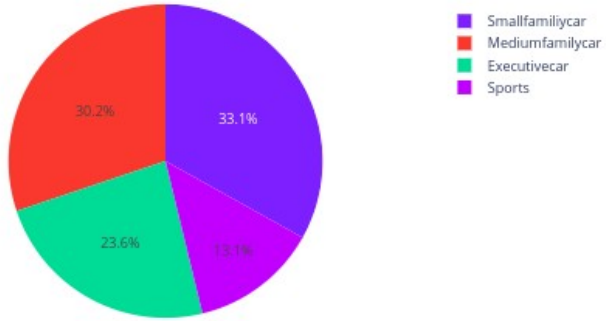


Average Vehicles Sold by Vehicle Type in the year 1985



Total Monthly Automobile Sales

Total Advertisement Expenditure by Vehicle Type in the year 1985



Back

Graded Quiz: Creating Dashboards with Plotly and Dash

Graded Quiz • 30 min

Due Oct 22, 11:59 PM IST

Congratulations! You passed!

Grade received 100% **Latest Submission Grade** 100% **To pass** 70% or higher

Retake the assignment in **7h 56m**

Go to next item

1. With Plotly, where can created web-based visualizations be displayed? **1 / 1 point**

- ○ Python
- ◉ Jupyter notebook
- ○ Dash
- ○ Matplotlib

Correct

Correct! Web-based visualizations created using Plotly Python can be displayed in Jupyter Notebook, saved to standalone HTML files, or served as part of pure Python-built web applications using Dash.

2. True or False. The Plotly graph objects module provides an automatically generated hierarchy of classes. It is the low-level interface to figures, traces, and layouts. **1 / 1 point**

Matplotlib

Correct
Correct! Web-based visualizations created using Plotly Python can be displayed in Jupyter Notebook, saved to standalone HTML files, or served as part of pure Python-built web applications using Dash.

2. True or False. The Plotly graph objects module provides an automatically generated hierarchy of classes. It is the low-level interface to figures, traces, and layouts. **1 / 1 point**

- ○ False
- ◉ True

Correct
Correct! The Plotly graph objects module provides an automatically generated hierarchy of classes. It is the low-level interface to figures, traces, and layouts.

3. Fill in the blank. A Plotly. graph contains a _____ object which has a dictionary structure. **1 / 1 point**

- ○ HTML
- ○ Matplotlib
- ◉ JSON

3. Fill in the blank. A Plotly. graph contains a _____ object which has a dictionary structure. **1 / 1 point**

- ○ HTML
- ○ Matplotlib
- ◉ JSON
- ○ Python

Correct
Correct! A Plotly. graph contains a JSON object which has a dictionary structure.

4. True or False. Real-time visuals on the dashboard simplify the comprehension of various aspects of the business. **1 / 1 point**

- ○ False
- ◉ True

Correct
Correct! Real-time visuals on the dashboard simplify the comprehension of various aspects of the business. Also, getting the big picture in one place can help

4. True or False. Real-time visuals on the dashboard simplify the comprehension of various aspects of the business. **1 / 1 point**

- ○ False
- ◉ True

Correct
Correct! Real-time visuals on the dashboard simplify the comprehension of various aspects of the business. Also, getting the big picture in one place can help businesses make informed decisions, thereby improving performance.

5. Fill in the blank. Matplotlib is a comprehensive library for creating static, animated, and interactive _____ in Python. **1 / 1 point**

- ○ Graphs
- ◉ Visualizations
- ○ Videos
- ○ Charts

Correct
Correct! Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

6. True or False. Data can be presented by using different types of dashboards. **1 / 1 point**

- ○ False
- ◉ True

Correct
Correct! Data can be presented by using different types of dashboards.

7. True or False. Plotly Express is a high-level wrapper for Plotly. **1 / 1 point**

- ◉ True
- ○ False

Correct
Correct! Plotly Express is a high-level wrapper for Plotly. It is a recommended starting point for creating the most common figures provided by Plotly because of its simple syntax. It uses graph objects internally.

8. What is a callback function? **1 / 1 point**

- (●) A decorator
- ( ) A line of Code
- ( ) A file
- ( ) An extension

Correct
Correct! Whenever there is a change in the input component value, the Callback function wrapped by the decorator is called, followed by the update to the output component children in the application layout.

9. True or False. Dash is an Open-Source User Interface Python library for creating reactive, web-based applications. **1 / 1 point**

- (●) True
- ( ) False

Correct
Correct! Dash is an Open-Source User Interface Python library for creating reactive, web-based applications. It is enterprise-ready and a first-class member of Plotly's open-source tools.

Correct
Correct! Dash is an Open-Source User Interface Python library for creating reactive, web-based applications. It is enterprise-ready and a first-class member of Plotly's open-source tools.

10. Which **two** parameters are needed to decorate a callback function? **Select two.**

1 / 1 point

- [x] Output

Correct
Correct! Output sets results returned from the callback function to a component id. The set input is provided to the callback function to a component id.

- [ ] Dash core
- [x] Input

Correct
Correct! Output sets results returned from the callback function to a component id. The set input is provided to the callback function to a component id.

- [ ] Dash

Course Summary

In this course, you learned how to use Data Visualization with Python to create charts and graphs for business or educational purposes. Graphs and charts are essential for communicating statistical information directly and efficiently. In this course, you also learned the history of Data Visualization with Python, why it was created, and who created it. Understanding the creation process of Data Visualization applications and programs is essential for understanding its uses, abilities, and importance. You can apply these skills to other graphing and statistical visualization applications that utilize Python and code.

Next Steps

After completing this course, you could take similar courses related to Data Visualization with Python, “Python for Data Science AI & Development, Python for Data Science, and Databases and SQL for Data Science with Python courses.”

Completed

Go to next item